Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 57 (2015) 1235 – 1241

Procedia Computer Science

3rd International Conference on Recent Trends in Computing 2015 (ICRTC-2015)

An optimized encryption technique using an arbitrary matrix with probabilistic encryption

Paresh Ratha[a], Debabala Swain[b*], Bijay Paikaray[c],Subhadra Sahoo[d]

[a]*Paresh Ratha, Dept. of CSE, CUTM*
[b]*Debabala Swain, School of Computer Engeering, KIIT University*
[c*]*Bijay Paikaray, Dept. of CSE, CUTM*
[d]*Subhadra Sahoo, M.Tech Scholar, CET, Bhubaneswar*

Abstract

This paper presents a simple Encryption/Decryption technique which enables all kinds of file for encryption and decryption. The method of encryption is not only simple, but also secured enough for transmission. The technique uses simple key generation using an arbitrary matrix. The final encryption is performed through a set of operations using the generated key and original data block and vice-versa in decryption. The generation of keys along with encryption and decryption is performed by the algorithm by putting in the data block and arbitrary matrix. The encryption process can be done by modifying the arbitrary matrix and multiple key can be generated for a single data block.

Peer-review under responsibility of organizing committee of the 3rd International Conference on Recent Trends in Computing 2015 (ICRTC-2015)

Keywords- Text Encryption, DES, AES, BLOWFISH, Arbitrary matrix, Multiple Key-Cipher generation.

1. Introduction

Encryption is an effective approach to attain security of data from unauthorized attack. It conceals the original information which can only be improved using a decryption process. In encryption the original data is passed through a series of procedure like, substituting, shifting and various mathematical processes to generate in a different form called cipher text. Key-based algorithms use an encryption key to encrypt the message [7]. The key-based encryption can be Symmetric Encryption which uses a single key to encrypt and decrypt the message or Asymmetric Encryption which uses two different keys – a public key to encrypt the message, and a private key to decrypt it [7]. There are several key based Encryption algorithms such as: DES, RSA, PGP, Elliptic curve, etc.[7] In this paper we have introduced a new probabilistic encryption technique using an arbitrary matrix with minimized complexity analysis and protected data extraction with enhanced security aspects.

* Corresponding author.
E-mail debabala.swain@gmail.com

Peer-review under responsibility of organizing committee of the 3rd International Conference on Recent Trends in Computing 2015 (ICRTC-2015)
doi:10.1016/j.procs.2015.07.422

2. Related Works.

2.1 LEA: Link Encryption Algorithm Proposed Stream Cipher Algorithm [1]

It is a stream cipher algorithm, which uses an 8-bit ASCII character for encryption and decryption. These characters combine with the plain text by bitwise addition to produce the cipher character. The algorithm is offering highest security level achieved through the high nonlinear complexity and the complete re-initialization before every encryption process [1]. This algorithm forms the key sequence by combining the two different sequences, where the first one has probable long period and the second one is high complexity by using nonlinear functions. Moreover, both of these sequences are statistically flat. The good statistical properties of the first sequence are in principle that of Liner Feedback Shift Registers (LFSRs) with primitive characteristic polynomials to achieve maximal-length period. Also, those of the second part are achieved by using well distributed generated substitution boxes, confusion, and diffusion. The LEA encryption algorithm can be divided into a driving part and a combining parts. The driving part consists of a set of maximum length Linear Feedback Shift Registers. It mainly governs the state sequence of the generator and is responsible for providing sequences of large periods and good statistics. The combining part is essentially nonlinear. It has the task to make the cipher stream generation to be mathematically complex.

2.2 The Hybrid Encryption Algorithm in Software Security [2]

The Hybrid encryption algorithm has developed a new fusion algorithm by combining Vigenere encryption algorithm and the Base64 encryption algorithm [2]. In the implementation part first, the improved Vigenere encryption algorithm, and then the Base64 encryption algorithm [2]. The first encryption algorithm generates a cipher text by inputting a plain text. Second, a key needs to be defined for the improved Vigenere encryption algorithm. Now using this key a second cipher text will be formed. Finally, using the Base64 encryption algorithm the last cipher text will be constructed which is to be sent. In the Decryption process the reverse must be followed to recover the original text.

2.3 Symmetric Encryption Algorithm for MANETs [3, 8]

During an encryption/decryption process a large amount of computing resources like CPU time, battery power and memory is consumed from the devices. In some networks like mobile ad hoc networks, as there is a constraint on power consumption, so there is a need to improvise the battery technology. So, it is essential to find a way to reduce the consumption of battery powered devices[3, 8]. This algorithm presents a potential solution of energy consumption in various handheld devices using commonly used symmetric key encryption algorithms. [3, 8] It was seen in Triple-DES after 600 encryptions using a 5 MB file the remaining battery power is 45%, which prevents the device from subsequent encryptions [3, 8]. In case of AES it was faster and efficient than other encryption algorithms. By enhancing the key size of 64 bits in AES the energy consumption has increased 8% without transferring any data. Reducing the number of rounds leads to power savings, but it makes the protocol insecure for AES.

3. Proposed Work

1. The proposed algorithm applies an arbitrary matrix key which gets multiplied by an initial vector, then subsequent substitute functions and conversions generate a sequence.
2. This sequence is used to be by both the sender and receiver with one at a time for encryption and decryption in each character.

3. In the encryption process the text characters enter continuously, producing cipher characters one at a time and vice-versa in the decryption at the receiver end. Proposed technique has the significance of generating poly alphabet ciphers with probabilistic encryption.

3.1 Algorithm for Key set generation

1. Consider the sequence for 0 to n-1 values for a source text of size n = pow (p, p) characters.
2. Convert each element of the sequence into a form with base p.
3. Represent the values of step 2 in a matrix form A of order n x p.
4. Subtract 1 from each element of A.
5. Consider a random matrix B of size p x p.
6. Multiply the matrix A with B to generate a result matrix R.
7. Substitute all positive integers of R with +1, negative integers to -1 and zero by 0 using a substitute function.
8. Increment each element of R by 1.
9. Convert each row of R, from the base p to decimal to generate the key sequence set.

3.2 Proposed Encryption Algorithm

1. Let C_i be the plain text for i = 0 to n-1.
2. Let the numerical equivalent NE_i of a character C_i.
   Where, N_i = ASCII value of C_i , for each i from 0 to n-1.
3. Find the sum index $S_i = NE_i + K_i$, for each i from 0 to n-1, where K_i is the i^{th} key generated in the sequence.
4. Find the remainder index $Ri = S_i \% 36$, for each i from 0 to n-1.
5. Find the cipher text for each input character as CT_i , where CT_i is the character equivalent of each R_i, where i varies from 0 to n-1.

3.3 Proposed Decryption Algorithm

1. Let CT_i be the cipher text.
2. Find the Alpha numeric equivalent NE_i for each CT_i .
   Where, N_i = ASCII value of C_i , for each i from 0 to n-1.
3. Find the sum index S_i, where $S_i = NE_i + 36$, if $0 \leq NE_i \leq 9$
   $= NE_i$, else.
4. Find the Difference index $D_i = S_i - K_i$, for i= 0 to n-1, where K_i is the i^{th} key generated in the sequence.
5. Find C_i, where C_i is the character equivalent of D_i, for i = 0 to n-1.

4. Test Results and Performance Analysis

4.1 Implementation Details

Algorithms are implemented using C programming code in Windows-7, 32 bit operating system on a 2.2 GHz Intel Core Duo processor using 2GB RAM to analyze their performance.

4.2 Complexity Analysis of Key Generation Algorithm

In each step it performs n number of computations, so the total number of computations is in multiples of n. So the time complexity of the proposed technique can be expressed as **O (n)**.

4.3 Avalanche effect

In Test-1, we have generated 256 key sequences using an arbitrary matrix MATRIX 1 of order 3X3.In Test-2 we have incremented 1 to each element of first column in MATRIX 2. So there was a variation of 19 keys in the sequence. In Test-3 we have decremented 1 to each element of first column in MATRIX 3. So there was a variation of 36 keys in the sequence.

From the above test it was identified that a little variation in the arbitrary matrix has a significant effect on the generated key sequence. The Cipher text is dependent on the generated key sequence. So a small variant in the key sequence has a relative consequence on the cipher text, which provides maximum avalanche effect. It provides maximum strength and security to the algorithm. It can be easily revealed from the Table-1.

Table 1. Results of Avalanche Test

Test No	No of modified Keys / Generated Keys	Original Text	Cipher Text
1	0/256	abcdef	PQRSBU
2	19/256	abcdef	PHRSBU
3	36/256	abcdef	PZR1BU

4.4 Execution Time Analysis

The nine text files of different sizes are used to carry out the tests, where an evaluation of three different algorithms AES, DES and Blowfish are performed. The experiments are conducted on the test system. The performance of these algorithms is evaluated based on parameters like, execution time, throughput and Avalanche effect.

Table 2.Execution Time Analysis of all four algorithms

INPUT SIZE (in KB)	EXECUTION TIME(in Sec)			
	DES	AES	BLOWFISH	PROPOSED
20	16.2	40.5	10.07	12.91
36	28.4	71.07	17.67	18.62
45	36.2	90.63	22.53	24.61
59	47.2	118.17	29.37	33.94
69	54.8	137.3	34.1	40.24
137	108.2	271.1	67.33	84.32
158	125.2	313.8	77.93	98.38
166	131	328.4	81.57	103.19
191	150	377.8	93.8	119.32
Total Execution Time	697.2	1748.77	434.37	535.53
Average Execution Time	77.46	194.30	48.26	59.50

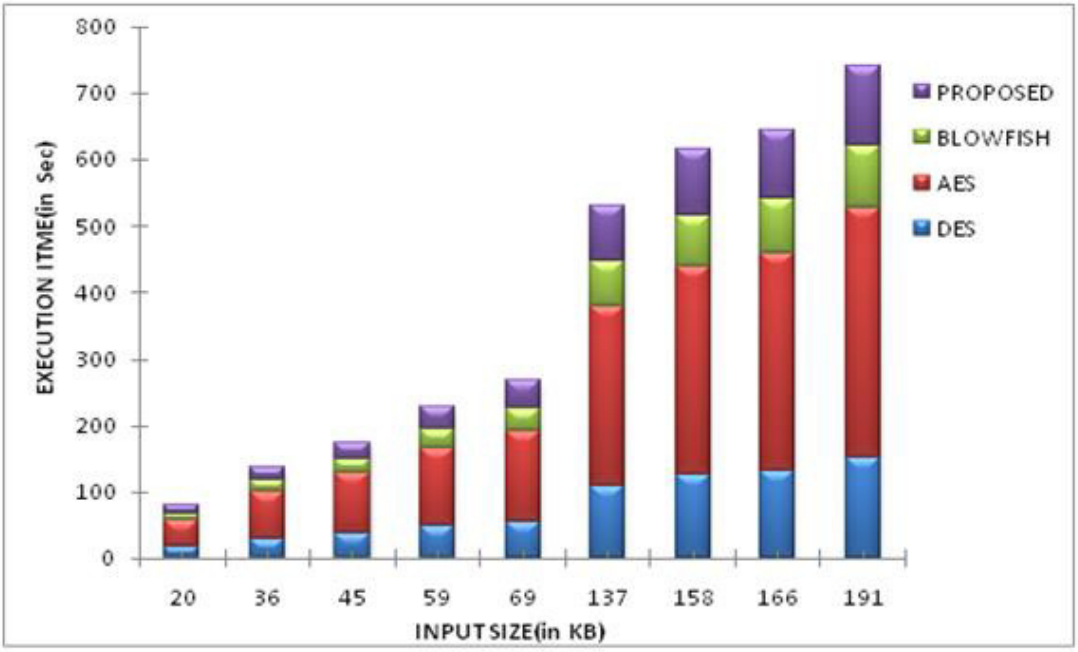

Fig 1. Execution time analysis between DES, AES, BLOWF1SHand PROPOSED algorithm

Fig 2. Throughput Analysis between DES, AES, BLOWFISH and PROPOSED algorithm

4.5 Throughput Analysis

Now the throughput of individual algorithm can be calculated by dividing the total plaintext size in Kilo bytes to the average execution time in seconds [3, 8]. Throughput of the algorithm is expressed in terms of kilobytes per second. As throughput is inversely proportionate to the consumed power of the algorithm, so it behaves as a major power saver. The overall performance of all three algorithms along with the proposed algorithm is given in Table 2 and figure-1.

From the Table-3, it is clear that the proposed algorithm has improved performance than AES, DES. It performs nearly 1.3 times faster than DES, 3 times faster than AES but BLOWFISH is the fastest and performs 1.25 times faster than PROPOSED algorithm. From the measured throughput figures AES is slowest and BLOWFISH algorithm is fastest.

Table-3. Throughput Analysis of DES, AES, BLOWFISH, PROPOSED Algorithm

Algorithm	Throughout (in KB/sec)
DES	1.263
AES	0.503
BLOWFISH	2.028
PROPOSED	1.645

4.6 Security Analysis

In this model we use a substitution function to generate the sequence. The substitution function substitutes all positive values to +1, negative values to -1, and zero with 0.This sequence is converted to equivalent character to generate the cipher text. So it is unfeasible to produce the arbitrary matrix from the cipher texts. Hence it is superior from other crypto algorithms. The complete Key sequence cannot be recovered; therefore the entire text cannot be retrieved.

5. Conclusion

In information Technology where security is the major apprehension, there encryption algorithm plays a key role. In this paper we have implemented the foremost algorithms, like DES, AES and BLOWFISH and evaluated their performance along with the PROPESED algorithm by considering different parameters such as: execution time, throughput and Avalanche Effect. Our future work will direct more towards analysis of security and performance issues of the PROPOSED algorithm using heterogeneous data types, like image encryption and video encryption.

References

[1] LEA: Link Encryption Algorithm Proposed Stream Cipher Algorithm, Ain Shams Eng J (2014), http://dx.doi.org/10.1016/j.asej.2014.08.001,www.elsevier.com/locate/asej

[2] The Application of Hybrid Encryption Algorithm in Software Security, Fourth International Conference on Computational Intelligence and Communication Networks (CICN), 2012 http://ieeexplore.ieee.org/xpl/abstractAuthors.jsp?arnumber=6375216.

[3] Evaluation of Symmetric Encryption Algorithms for MANETs, Dec. 2010 IEEE International Conference on Computational Intelligence and Computing Research (ICCIC). http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=5705754

[4]Baocang Wang, Qianhong Wu, Yupu Hu: A Knapsack Based Probabilistic Encryption Scheme, On Line March 2007.

dl.acm.org/citation.cfm?id=1279149.

[5] Beth T. and Gollmann D. "Algorithm Engineering for Public Key Algorithms". IEEE Journal on Selected Areas in Communications; Vol. 7, No 4, PP. 458-466, http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=17708, July 2003.

[6] Performance Analysis of Encryption Algorithms for Information Security, 2013 International Conference on Circuits, Power and Computing Technologies (ICCPCT-2013), pp 840-844.

[7] Majdi Al-qdah, Lin Yi Hui, Simple Encryption/Decryption Application, Volume - 1 Issue - 1, pp-33-40, http://www.cscjournals.org/library/manuscriptinfo.php?mc=IJCSS-4#MCAI .

[8] Levent Ertaul, Nitu J. Chavan , Elliptic Curve Cryptography based Threshold Cryptography (ECC-TC) Implementation for MANETs, Vol. 7 No. 4 pp. 48-61, http://paper.ijcsns.org/07_book/html/200704/200704007.html